Illinois Institute of Technology Department of Computer Science

Midterm 1

CS 440— Programming Languages
Spring 2006
February 24, 2006 13:50–15:05

This is a **closed book** and **closed notes** exam.
You are **not** allowed to use calculators or computers during this exam.
Do **ALL** problems in this booklet. Read each question very carefully.
You may detach pages, but **you must return all pages of this exam.**

Name	
Email ID	@iit.edu

Do **not** place your social security number anywhere on this exam.

Problem	Points	Score
1	6	
2	6	
3	4	
4	6	
5	4	
6	4	
7	6	
8	6	
9	3	
10	3	
11	3	
12	3	
13	6	
14	6	
15	6	
16	6	
17	6	
Total	84	
Percent	100	

Recursion

Use recursion to write the following functions. You do **not** have to use tail recursion, but you may do so if you want. You are **always** allowed to make helper functions.

Question 1) (6 points) Write a recursive function `sumonen : int -> int` that takes an integer n and returns the sum $\Sigma_{i=1}^{n} i$. If $n < 0$ simply return 0.

```
# let rec sumonen n = ...
val sumonen : int -> int = <fun>
# sumonen 10;;
val - : int = 55
```

Question 2) (6 points) Write the function `listPlusFour xx : int list -> int list`, which outputs a list of the input list's elements incremented by two.

```
# let rec listPlusTwo xx = ...
val listPlusTwo : int list -> int list = <fun>
# listPlusTwo [2;4;0;-2;-1];;
- : int list = [6; 8; 4; 2; 3]
```

Tail Recursion

Question 3) (4 points) What is the advantage of tail recursion over non-tail recursion?

Question 4) (6 points) Write the function `digits xx : int list -> int`, which computes $a_0 10^0 + a_1 * 10^1 + \cdots + a_n * 10^n$ from the list $[a_0; a_1; \ldots; a_n]$. Use tail recursion. You may write a helper function, or use standard library functions if you want.

```
# let digits xx = ...
val digits : int list -> int = <fun>
# digits [8;6;7;5;3;0;9];;
- : int = 8675309
```

Higher Order Functions

Question 5) (4 points) Write the code for `map : ('a -> 'b) -> 'a list -> 'b list`.

Question 6) (4 points) Write the code for `fold_right: ('a -> 'b -> 'b) -> 'a list -> 'b -> 'b`.

Question 7) (6 points) Using either `map` or `fold_right`, write the function `digits xx : int list -> int`, which computes $a_0 10^0 + a_1 * 10^1 + \cdots + a_n * 10^n$ from the list $[a_0; a_1; \ldots; a_n]$. You may **not** use explicit recursion. (You may use `List.rev`, however.)

```
# let digits xx = ...
val digits : int list -> int = <fun>
# digits [8;6;7;5;3;0;9];;
- : int = 8675309
```

Question 8) (6 points) Using either `map` or `fold_right`, write the function `decList xx : int list -> int list`, which decrements each element of a list. You may **not** use explicit recursion.

```
# let decList xx = ...
val decList : int list -> int = <fun>
# decList [2;4;6];;
- : int list = [1;3;5]
```

Types

Question 9) (3 points) What is the type of the following function?

```
let foo x = x * x
```

Question 10) (3 points) What is the type of the following function?

```
let bar y = [3.18] @ y
```

Question 11) (3 points) What is the type of the following function?

```
let baz z = 10 + (z 5)
```

Question 12) (3 points) What is the type of the following function?

```
let gop w = w (w 2)
```

Question 13) (6 points)
The following type creates a list-like data-structure.

```
type tsil = Snoc of tsil * int
          | Lin
```

Write a function `sumTsil : tsil -> int` which returns the sum of all the elements of the `tsil`.

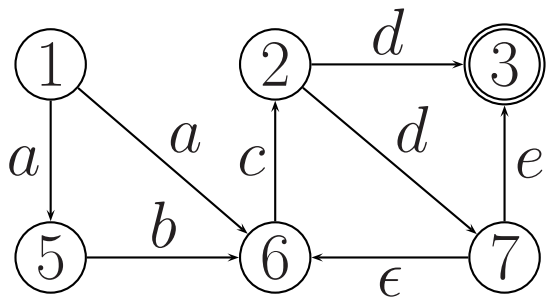
```
# let rec sumTsil xx =
val sumTsil : tsil -> int = <fun>
```

Question 14) (6 points)
Write a function `list2tsil : int list -> tsil` which converts an OCaml l ist into a `tsil`.

```
# let rec list2tsil xx = ...
val list2tsil : int list -> tsil
```

Automata

Question 15) (6 points) Convert the following NFA to a DFA. (State 1 is the start state.)



LL Grammars

Question 16) (6 points) In class we discussed that regular languages and context-free languages are not of equal power, but that one class is more capable than the other. Which language class is more expressive? Give an example of a grammar that is able to be expressed by one class of language but not the other.

Question 17) (6 points) What does it mean for a grammar to be *ambiguous*? Give an example of an ambiguous grammar, and prove its ambiguity.